Implémentation et illustration

« Le sage n'affirme rien qu'il ne puisse prouver. »

Proverbe latin

V.1. Introduction

Notre étude a permis de fournir une méthodologie permettant de prendre en considération les propriétés non-fonctionnelles depuis la modélisation des processus métier collaboratifs jusqu'à la réconciliation et sélection de services. Dans les chapitres précédents, nous avons présenté les différentes descriptions théoriques de nos cadres. Ce chapitre a pour objectifs de présenter, d'une part, l'implémentation de cette méthodologie afin qu'elle soit exploitable, et d'autre part la mise en œuvre de nos propositions sur un cas d'étude pour illustrer et valider nos travaux de recherche.

V.2. Plateforme A2S

Dans cette section, nous présentons la plateforme A2S (pour *Activities to Services*). Cette plateforme englobe les deux prototypes développés au cours de nos travaux :

- une plateforme de modélisation « *Petals BPM-NFR* » (niveau métier) qui se base principalement sur l'outil de modélisation BPMN 2.0 « Petals BPM » auquel nous avons implémenté une couche qui enrichit la modélisation des processus par l'annotation non-fonctionnelle ;
- une plateforme de gouvernance SOA « *EasierGov-NFR* » (niveau technique) qui s'intéresse à la rationalisation des services et de leurs propriétés non-fonctionnelles, et à la réconciliation non-fonctionnelle. Cette réconciliation hérite les résultats obtenus par la réconciliation fonctionnelle (effectuée par l'outil « EasierSBS »).

La Figure 51 présente une vue de haut niveau de l'architecture de la plateforme A2S. Ses composants seront détaillés dans les sections qui suivent.

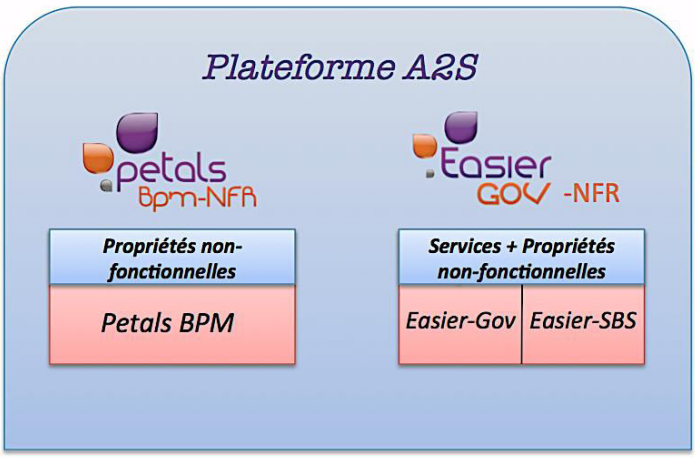

Figure 51 : *Architecture de la plateforme A2S*

V.2.1. Plateforme de modélisation : Petals BPM-NFR

Comme nous l'avons détaillé précédemment (cf. section III.2.2), Petals BPM-NFR représente notre plateforme graphique qui permet aux utilisateurs métier (architectes métier), d'une part, de modéliser les processus métier collaboratifs en suivant le standard BPMN 2.0, et d'autre par d'annoter les activités de ces processus par des exigences non-fonctionnelles. Cette annotation suit le standard WSQF pour la modélisation des propriétés non-fonctionnelles et permet également aux utilisateurs de définir leurs préférences (poids) par rapport à chacune des propriétés non-fonctionnelles choisies. La Figure 52 permet d'illustrer les fonctionnalités de notre plateforme Petals BPM-NFR.

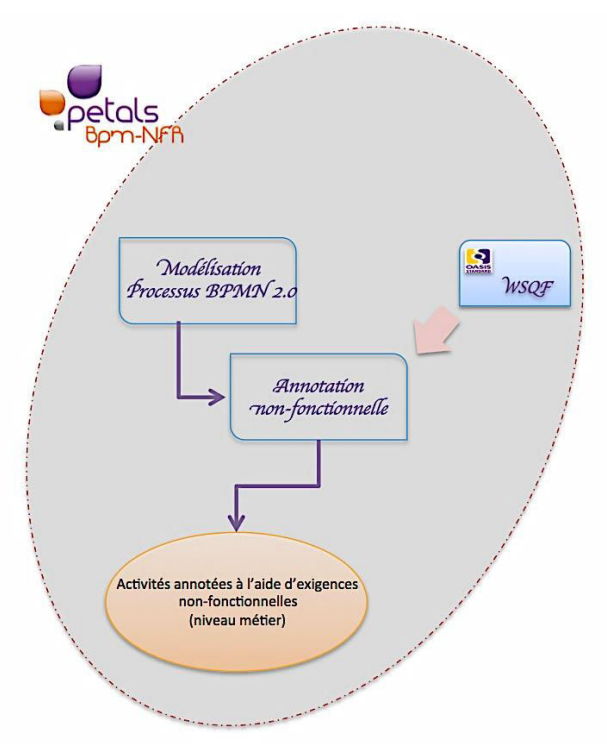

Figure 52 : *Fonctionnalités de Petals BPM-NFR.*

V.2.1.1. Architecture technique

Petals BPM-NFR est une application Client / Serveur et possède une architecture en couches qui se découpe en deux parties : (i) une partie client basée sur l'environnement GWT (nous présentons GWT dans la section qui suit) et (ii) une partie serveur entièrement écrite en Java. GWT propose un mécanisme d'appels de procédures distantes (*Remote Procedure Call* - RPC) qui permet la communication entre ces deux parties. Ce mécanisme de RPC utilisant la technologie de communication Ajax (pour *Asynchronous JavaScript and XML*) rend les appels de méthodes asynchrones. Cette architecture technique est schématisée par la Figure 53.

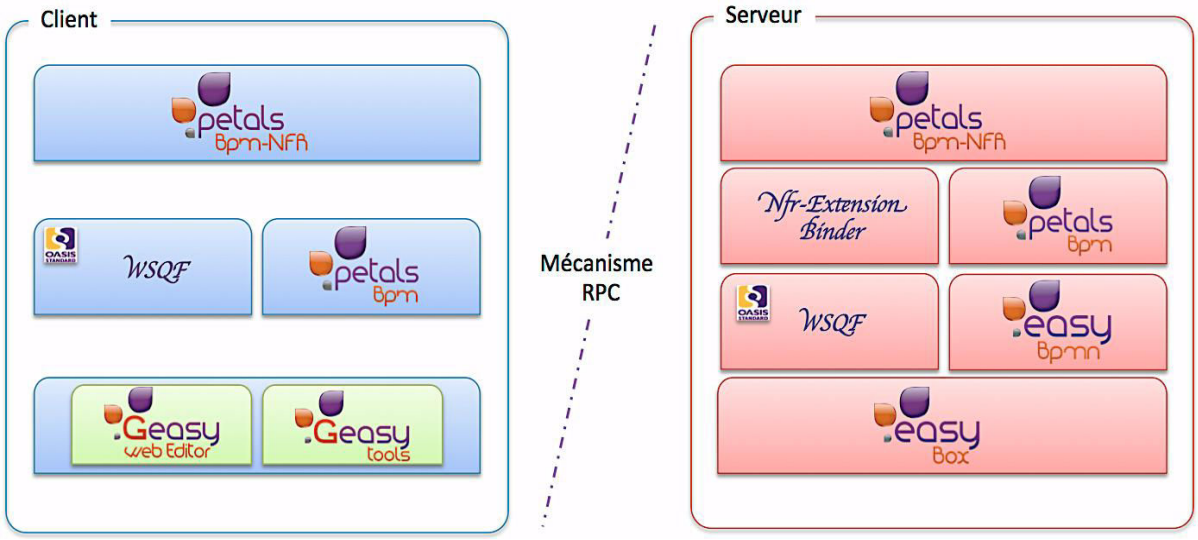

Figure 53 : *Architecture technique de Petals BPM-NFR.*

➢ *La partie client*

Petals BPM-NFR possède une interface utilisateur implémentée en GWT. Cette interface permet aux utilisateurs métier / architectes de modéliser graphiquement leurs processus, de les enrichir par des exigences non-fonctionnelles et d'exprimer leurs préférences par rapport à chacune des exigences non-fonctionnelles. La Figure 54 détaille les étapes de modélisation des processus métier collaboratifs annotés à l'aide d'exigences non-fonctionnelle en utilisant notre plateforme.

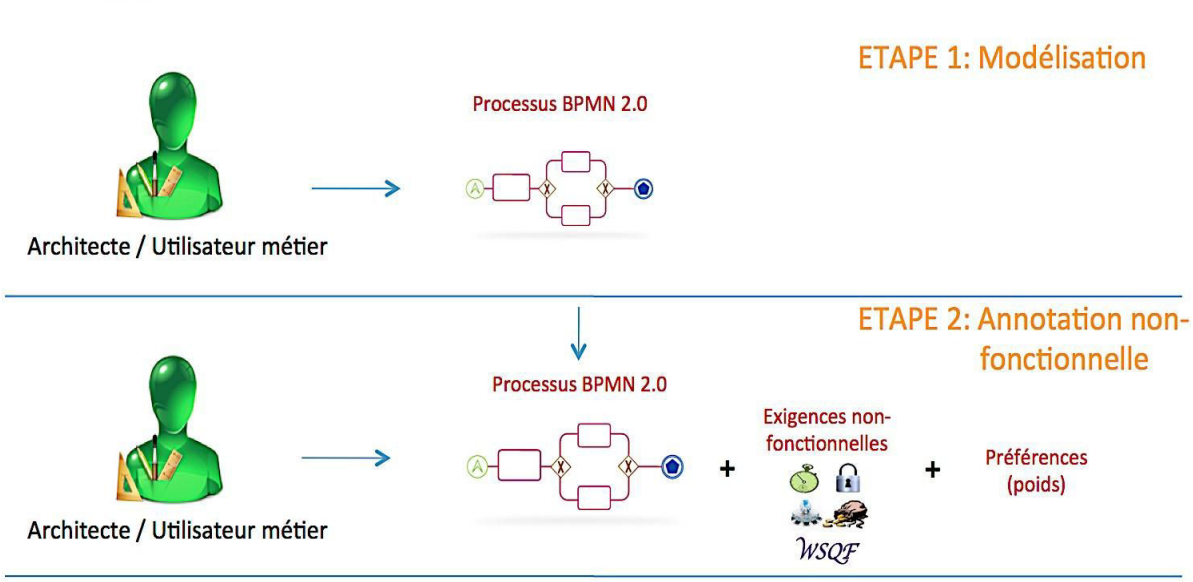

Figure 54 : *Modélisation des processus métier annotés avec Petals BPM-NFR (coté client).*

Petals BPM-NFR, du coté client, est basé sur le composant Petals BPM pour la modélisation graphique et sur le standard WSQF pour l'annotation non-fonctionnelle.

- *Petals BPM* : un outil Web open-source de modélisation graphique des processus métier. Il permet aux utilisateurs de créer en ligne des processus collaboratifs suivant le standard BPMN 2.0. Cette application est le résultat de l'utilisation des librairies GeasyWebEditor et GeasyTools ;
- *GeasyWebEditor* : une librairie basée sur le framework graphique GWT. Elle permet de faciliter la création de l'outil de modélisation graphique dans un environnement Web ;
- *GeasyTools* : un ensemble de librairies graphiques GWT qui permet de créer des applications riches et interactives. Ces librairies proposent un ensemble de composants graphiques de base qui seront réutilisables dans tout le projet.

➢ *La partie serveur*

Petals BPM-NFR du coté serveur possède des librairies personnalisées écrites en Java et basées sur Petals BPM et NFR Extension Binder.

- *Petals BPM* : la partie serveur de Petals BPM permet de faire les modèles sous format de fichiers. Elle permet également d'exporter / importer les modèles vers / depuis le format BPMN 2.0 ;
- *EasyBPMN* : une librairie qui correspond au modèle BPMN 2.0 traité par EasyBox. Elle permet également de gérer la transformation de BPMN 2.0 en un processus exécutable BPEL [Rajsiri et al., 2011] ;
- *NFR Extension Binder* : une librairie qui s'intègre au serveur Petals BPM afin de permettre d'exporter / importer les extensions non-fonctionnelles au sein des fichiers BPMN 2.0 ;
- *WSQF* : une librairie qui correspond au modèle Java du standard WSQF écrit par EasyBox ;
- *EasyBox* : une librairie qui permet de transformer des schémas XML (eXtensible Markup Language) en modèles de classes Java équivalentes et vice versa (sérialisation et désérialisation avancées). En effet, cette librairie facilite l'utilisation des fonctionnalités Jaxb (Java Architecture for XML Binding).

V.2.1.2. Implémentation avec GWT et Java

Pour implémenter les différentes fonctionnalités de notre modeleur Petals BPM-NFR, nous avons choisi l'environnement GWT (*Google Web Toolkit*) pour la modélisation graphique des processus et des exigences non-fonctionnelles. Nous avons également choisi le langage Java pour

effectuer la transformation des schémas de données en modèles de classes, ainsi que le traitement et la génération des fichiers BPMN 2.0.

La technologie GWT [Dewsbury, 2007] a été développée en 2006 par Bruce Johnson et lancée en open-source en 2007. GWT est un ensemble d'outils permettant, d'une part, de créer et d'optimiser des applications Web complexes, riches, en utilisant Java comme langage de développement et d'autre part, de compiler ce code en JavaScript et HTML. Ce code est interprétable automatiquement par les principaux navigateurs Web (*Google Chrome*, *Mozilla Firefox*, etc.). Par ailleurs, GWT permet aux développeurs de créer leurs propres composants graphiques, appelés *widgets* qui peuvent être facilement réutilisables et intégrés dans plusieurs projets.

Dans le cadre de la réalisation de Petals BPM-NFR, nous avons implémenté un ensemble de librairies qui couvrent les fonctionnalités suivantes :

- Prise en considération des exigences non-fonctionnelles : nous avons implémenté un ensemble de librairies permettant d'annoter non-fonctionnellement les processus métier collaboratifs. Ces librairies concernent : (i) la transformation (EasyBox) du schéma de données du standard WSQF en des modèles de classes Java équivalents. Cette transformation permet de manipuler les propriétés non-fonctionnelles en tant qu'objets Java, (ii) l'extension du modèle BPMN 2.0 (fourni par Petals BPM) en rajoutant les objets « propriétés non-fonctionnelles » ;
- Import direct dans la plateforme graphique de diagrammes BPMN 2.0 annotés non-fonctionnellement : Petals BPM-NFR permet aux utilisateurs d'importer (*uploader*) des fichiers BPMN 2.0 annotés par des exigences non-fonctionnelles (ceci est seulement dans le cas où cette annotation respecte le modèle WSQF). Toutes les informations que ces fichiers contiennent et qui sont supportées par Petals BPM-NFR sont mises à disposition pour la modélisation en cours ;
- Export de diagrammes modélisés et annotés sous format BPMN 2.0 : Petals BPM-NFR permet aux utilisateurs d'exporter leurs processus modélisés et annotés par des exigences non-fonctionnelles sous forme de fichier BPMN 2.0. Cet export est possible si celui-ci passe une étape de validation. Deux niveaux de validation sont pris en compte : (i) une validation de conformité à la norme BPMN 2.0 et (ii) une validation de la conformité des types de valeurs des propriétés non-fonctionnelles à la norme WSQF. Le premier niveau de validation a été réalisé par Petals BPM et nous le réutilisons durant nos travaux ;
- Modélisation graphique : une interface utilisateur (*User Interface* - UI) en GWT a été développée pour Petals BPM-NFR. Elle enrichit l'interface de Petals BPM en ajoutant une partie dédiée à l'annotation graphique des activités par des exigences non-fonctionnelles (dans le respect de la spécification de WSQF). Nous avons également implémenté les

modèles internes relatifs à cette interface qui permettent d'enrichir la palette des composants BPMN 2.0 et de la lier à la zone dédiée aux propriétés dans le but de faciliter aux utilisateurs métier la modélisation. Ainsi, lors de la modélisation, à chaque fois que l'utilisateur fait un glisser – déposer (*drag and drop*) d'une activité BPMN 2.0 dans la zone de modélisation, l'interface dédiée à l'annotation non-fonctionnelle apparaît dans la zone des propriétés. La Figure 55 représente une capture d'écran de l'interface utilisateur de l'éditeur Petals BPM-NFR.

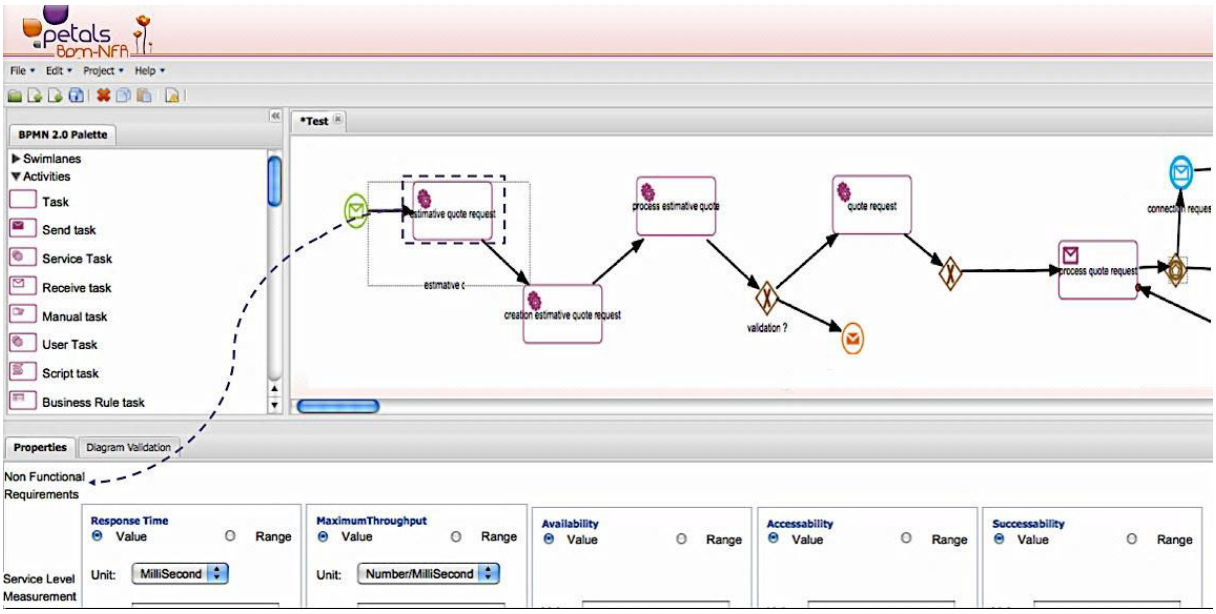

Figure 55 : *Capture d'écran de la plateforme Petals BPM-NFR.*

V.2.2. Plateforme de gouvernance SOA et de réconciliation : EasierGov-NFR

Comme nous avons pu le voir dans les chapitres précédents, un grand nombre de services, d'activités métier et de propriétés non-fonctionnelles est à prendre en charge. Nous souhaitons par le prototype EasierGov-NFR rationaliser la gestion des services et de leurs propriétés non-fonctionnelles et fournir le moteur de réconciliation non-fonctionnelle. Pour y parvenir, nous avons fortement basé les choix des composants d'EasierGov-NFR sur des standards (cf. Chapitre III). Nous avons également défini un ensemble de librairies implémentées avec le langage Java qui permettent de gérer les fonctionnalités d'EasierGov-NFR. Chacune de ces librairies expose ces fonctions dans une interface dédiée, sous la forme d'une API (*Application Programming Interface*) Java (*-api) qui sera implémentée par la suite (*-impl).

V.2.2.1.Rationalisation des services et de leurs propriétés non-fonctionnelles

EasierGov-NFR fournit deux APIs « Services API » et « SLA-API ». La première est basée sur la spécification WSDL pour la description de services, et la deuxième suit les standard WS-Agreement et WSQF.

Services API *:* cette librairie fournit une interface que nous appelons « *Services API* » (cf. Figure 56).

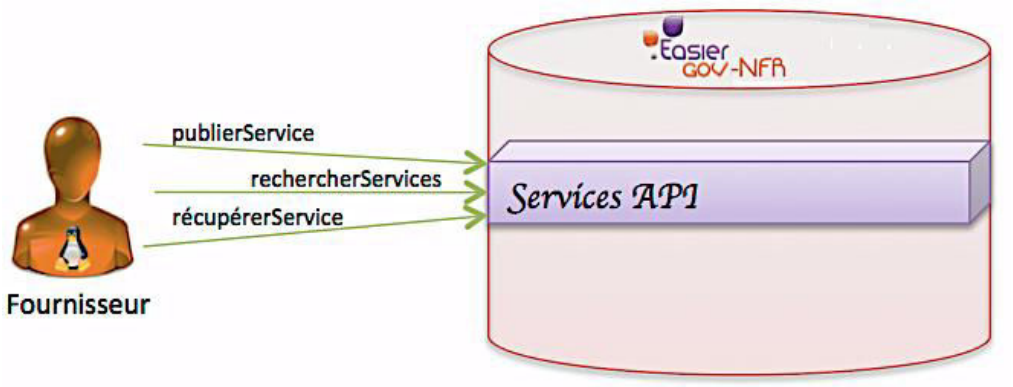

Figure 56 : *Principales méthodes de la librairie « services API ».*

Elle contient un ensemble d'opérations permettant de gérer les services. Les principales méthodes de cette librairie sont les suivantes :

- publierService (WSDL) : publier un nouveau service Web dans le registre à partir de la description de services passée en paramètre d'entrée. Cette description respecte les règles définies dans le standard de services WSDL ;
- Liste <ServiceId> rechercherServices (query) : rechercher une liste d'identifiants (id) de services à partir des paramètres d'entrées ;

- Service recupererService (idService) : obtenir la description de services (WSDL) à partir de l'identifiant demandé.

Cette librairie permet également de gérer d'autres méthodes pour les services telles que la récupération et la manipulation de l'interface du service et de son point d'accès (*endpoint*).

SLA API *:* Interface fournit un ensemble de méthodes permettant de créer et de manipuler les modèles de contrats de type SLA en respectant le standard WS-Agreement (cf. Figure 57) :

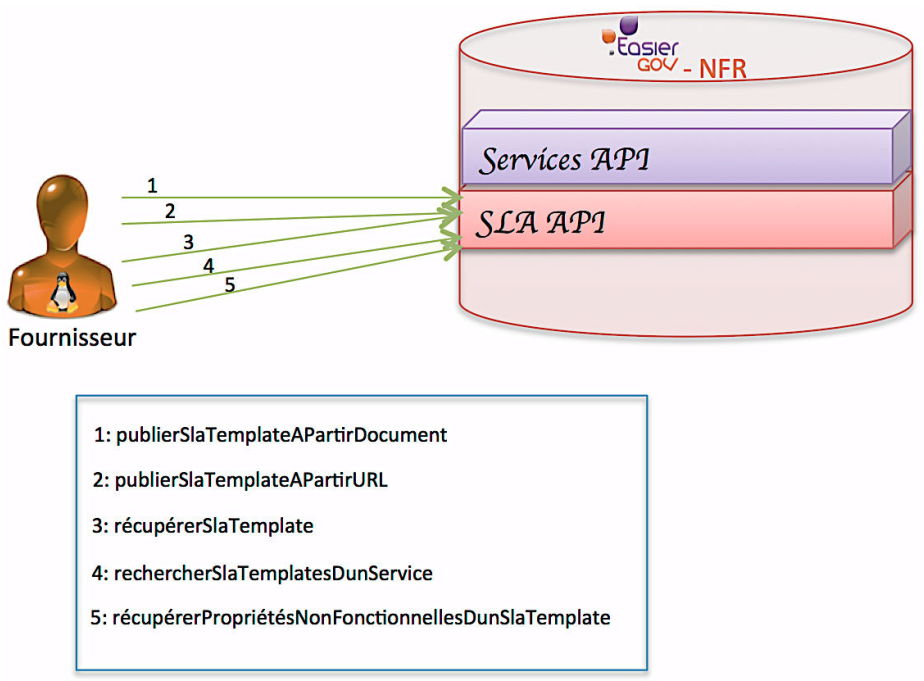

Figure 57 : *Principales méthodes de la librairie « SLA API ».*

- publierSlaTemplateAPartirDocument (wsagreement) : enregistrer un nouveau modèle de contrat dans le registre de gouvernance à partir du document passé en paramètre d'entrée. Ce document suit la description définie dans la norme WS-Agreement. Chaque modèle de contrat est relatif à un service défini dans le contrat par son adresse URL. Dans notre exemple (cf. Figure 58), le modèle SLA concerne le service pompier situé à l'adresse : esb://http://www.example.org/firemen//:firemen@firemenSOAP). Cette méthode permet également de parcourir le modèle de contrat, et d'enregistrer dans le registre de gouvernance, pour le service en question, les valeurs de toutes les propriétés non-fonctionnelles que le contrat contient ;

```
<wsag:Template wsag:TemplateId="AID1"
    xmlns:wsag="http://schemas.ggf.org/graap/2007/03/ws-agreement"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:wsad="http://www.w3.org/2005/08/addressing"
    xmlns:wsqdl="http://www.nca.or.kr/2006/wsqdl"
    xsi:schemaLocation="http://schemas.ggf.org/graap/2007/03/ws-agreement wsagreement10.xsd ">
    <wsag:Name>wsagreement</wsag:Name>
    <wsag:Context>
        <wsag:AgreementInitiator xsi:type="anyType" />
        <wsag:AgreementResponder xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:type="wsad:EndpointReferenceType">
            <wsad:Address>esb://http://www.example.org/firemen/::firemen@firemenSOAP</wsad:Address>
        </wsag:AgreementResponder>
```

Figure 58 : *SLA : adresse du service.*

- publierSlaTemplateAPartirURL (url) : publier un nouveau modèle de contrat SLA à partir de son adresse URL (pour Uniforme Ressource Locator). Tout comme la méthode de la publication à partir de la description du modèle du contrat, cette méthode permet également de parcourir, d'analyser et d'enregistrer les valeurs des propriétés non-fonctionnelles ;
- Template récupérerSlaTemplate (idTemplate) : obtenir la description du modèle de contrat SLA depuis son identifiant ;
- Liste < idTemplate > rechercherSlaTemplatesDunService (idService) : rechercher et retourner tous les modèles de contrat relatifs au service (selon son identifiant) défini dans le paramètre d'entrée ;
- Liste < idNFR > récupérerPropriétésNonFonctionnellesDunSlaTemplate (idTemplate) : rechercher et retourner l'ensemble des propriétés non-fonctionnelles de WSQF contenu dans le template SLA.

V.2.2.2.Moteur de réconciliation non-fonctionnelle

Ce module est dédié à la réconciliation entre le niveau métier et le niveau technique en se basant sur les critères non-fonctionnels. À partir du processus métier modélisé dans Petals BPM-NFR et annoté par des exigences non-fonctionnelles, ce moteur de réconciliation permet de trouver le service ou la composition de services qui se rapproche des exigences de l'activité métier. Nous détaillons dans la Figure 59 notre démarche globale permettant d'effectuer cette réconciliation.

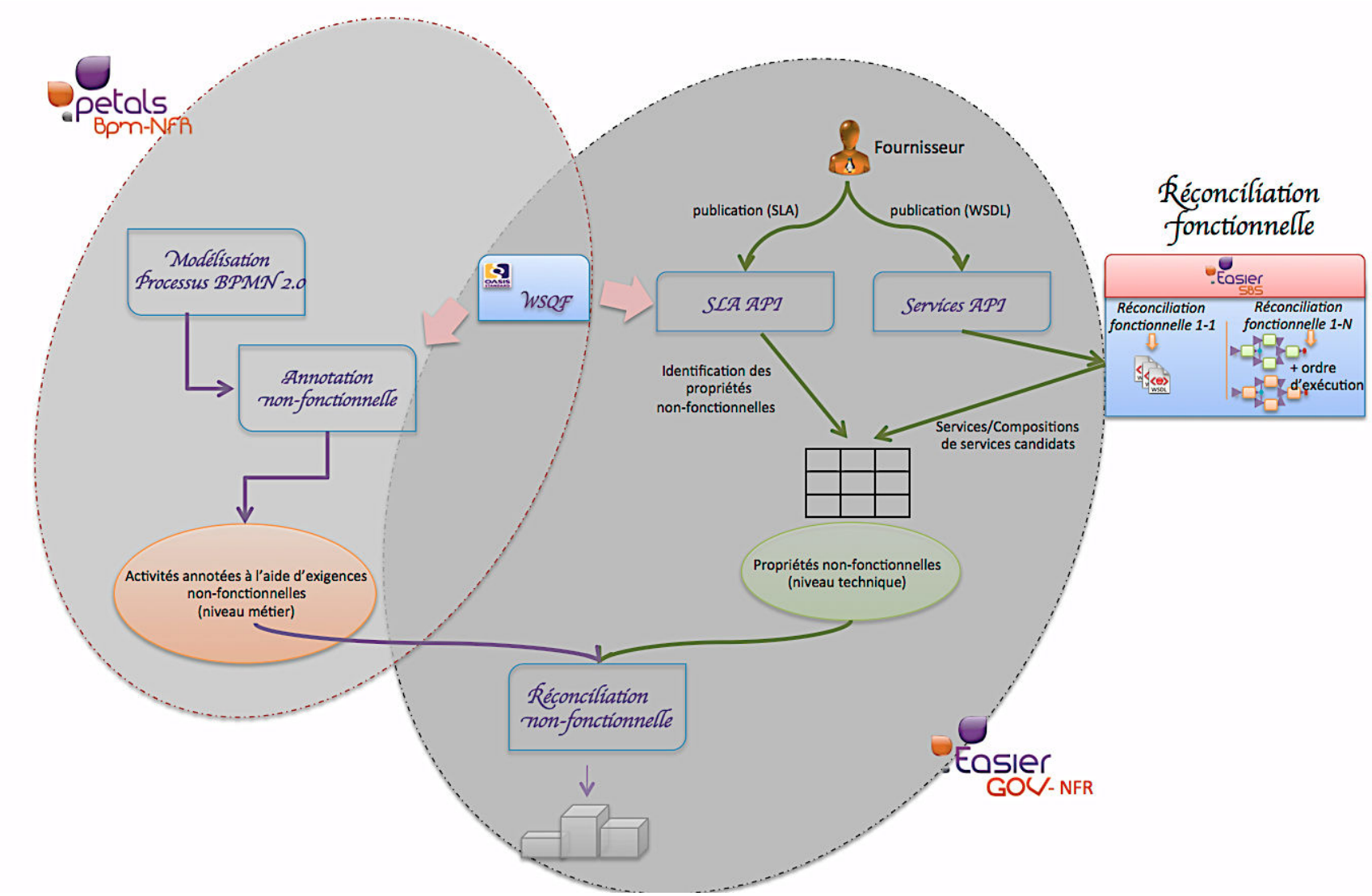

Figure 59 : *Moteur de réconciliation non-fonctionnelle.*

Ce moteur de réconciliation non-fonctionnelle utilise les librairies de services et de SLA détaillées précédemment, afin de permettre aux fournisseurs de services de publier dans un registre unique leurs services et de leur associer (en les publiant également) un ou plusieurs modèles SLA.

À partir des services publiés dans le registre, nous appliquons le filtre fonctionnel fourni par le composant EasierSBS établi dans [Boissel-Dallier, 2012]. Ce filtre fournit pour chacune des activités métier du processus une liste de services unitaires (dans le cas d'une réconciliation de type 1-1) ou un ensemble de compositions de services (dans le cas d'une réconciliation 1-N) ;

Pour chacune des activités métier et à partir des propriétés non-fonctionnelles enregistrées dans le registre EasierGov, nous construisons une matrice contenant les services et/ou les compositions de services et leurs propriétés non-fonctionnelles, en nous limitant à celles qui ont été choisies lors de l'étape de l'annotation du niveau métier.

À partir de ces propriétés non-fonctionnelles techniques et des exigences non-fonctionnelles modélisées au niveau métier Petals BPM-NFR, nous appliquons l'algorithme de réconciliation (cf. Chapitre IV) qui correspond au type de réconciliation fonctionnelle effectuée (1-1 ou 1-N). À l'issue de cette étape, nous obtenons pour chacune des activités métier, un classement non-fonctionnel des services. Le premier est celui qui se rapproche le plus des exigences de l'utilisateur définies dans l'activité métier.

V.2.3. Intégration dans une architecture SOA open-source

Les deux plateformes Petals BPM-NFR et EasierGov-NFR s'intègrent au sein de la gamme d'applications open-sources développées par l'équipe de recherche Petals Link de Linagora afin de permettre leur implantation dans une architecture réelle.

La plateforme de modélisation Petals BPM-NFR permet, à plusieurs utilisateurs métier, de créer des processus métier collaboratifs (suivant la norme BPMN 2.0) annotés à l'aide d'exigences non-fonctionnelles suivant le standard WSQF. Petals BPM-NFR est lié à la plateforme EasierGov-NFR pour gérer la réconciliation entre les activités métier et les services techniques disponibles (réconciliation fonctionnelle par le biais d'EasierSBS). Petals BPM-NFR est également lié à l'outil Petals BPM qui comprend une librairie permettant de créer, à partir des services, les processus BPEL correspondants. Ces processus sont exécutables dans l'infrastructure de services (le bus de services) Petals ESB.

La plateforme EasierGov-NFR est lié à l'outil de monitoring EasierBSM (*Business Service Monitoring*) [Lesbegueries et al., 2012] qui est également lié à Petals ESB afin de surveiller l'exécution des contrats de services et de notifier dans le cas d'une violation. La Figure 60 décrit l'intégration de Petals BPM-NFR et d'EasierGov-NFR dans l'architecture SOA existante.

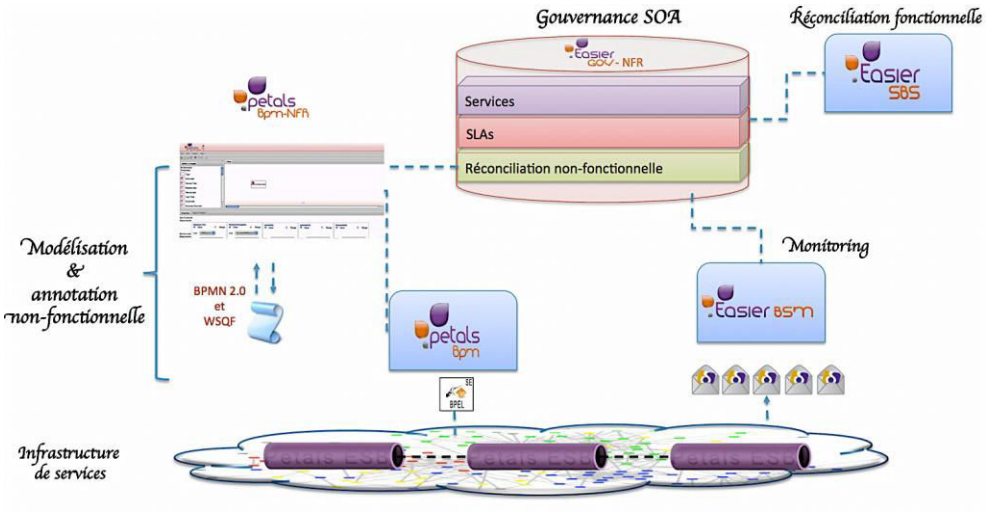

Figure 60 : *Intégration de Petals BPM-NFR et EasierGov-NFR dans une architecture SOA.*

V.2.4. Démarche de qualité constante

Les produits développés par l'équipe de recherche et développement Petals de Linagora respectent un ensemble de règles de développement et sont soumis à la démarche d'intégration continue. Au même titre que ces produits, les plateformes Petals BPM-NFR et EasierGov-NFR suivent le même principe d'intégration continue. Cela signifie qu'à chaque modification du code source ou d'une dépendance, les tests unitaires de chaque composant impliqué dans la plateforme en question sont relancés afin de vérifier qu'aucune régression fonctionnelle n'a eu lieu. Nous présentons dans les figures suivantes (cf. Figure 61 et Figure 62) les résultats des tests d'intégration continue respectivement pour Petals BPM-NFR et EasierGov-NFR sur la plateforme Jenkins[6].

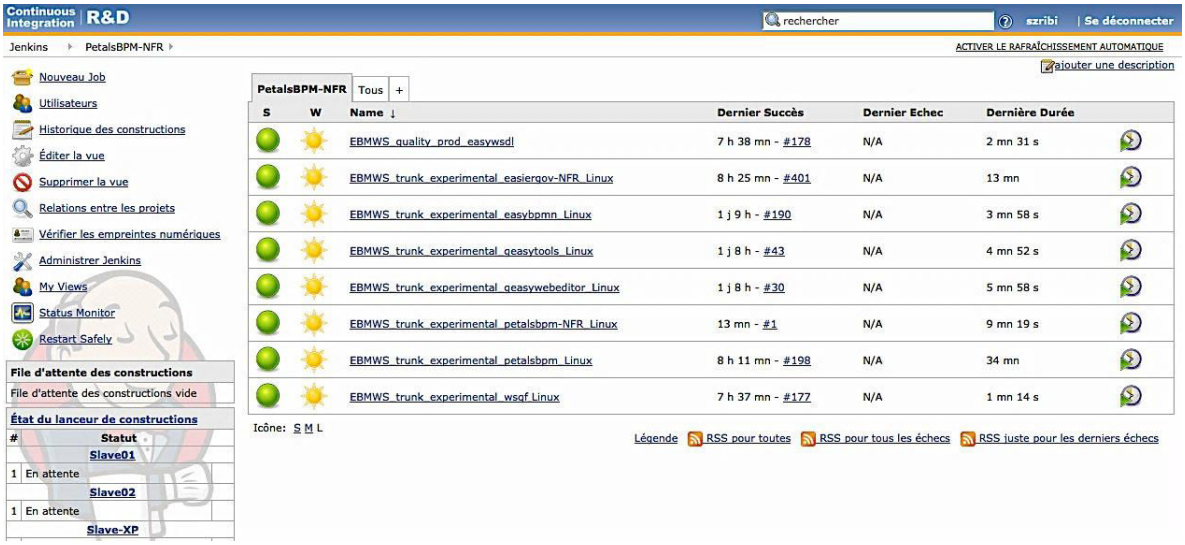

Figure 61 : *Résultats des tests d'intégration continue de Petals BPM-NFR.*

[6] Jenkins est un serveur open-source d'intégration continue pour les projets Java développés avec maven.

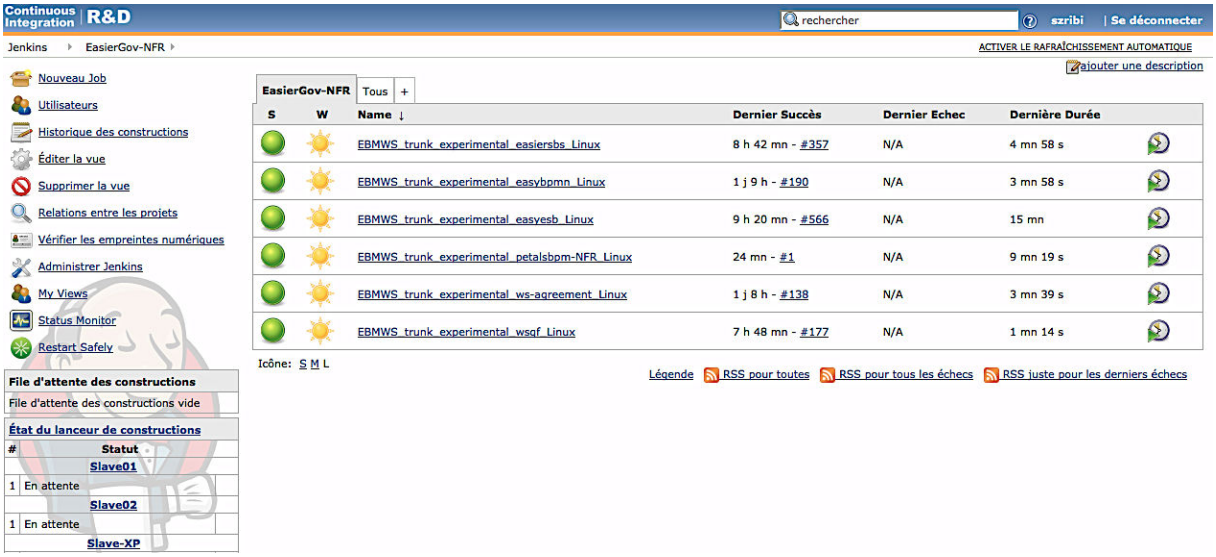

Figure 62 : *Résultats des tests d'intégration continue d'EasierGov-NFR.*

Comme nous pouvons l'observer, tous les voyants sont au vert, les deux plateformes Petals BPM-NFR et EasierGov-NFR assurent bien un état stable avec les autres produits Petals utilisés pour l'implémentation de chacune.

V.3. Illustration de la démarche proposée

Nous présentons dans cette section un cas d'étude fictif permettant d'illustrer l'applicabilité de notre méthodologie.

V.3.1. Présentation du cas d'étude

Sazri est une entreprise spécialisée dans la vente en ligne de chaussures. Elle souhaite trouver une entreprise de marketing prédictif avec laquelle elle envisage de collaborer afin d'améliorer ses ventes. Sazri lui fournirait la fiche du produit à promouvoir et son fichier clients. L'entreprise de marketing aurait pour rôle d'étudier les comportements clients afin d'anticiper leurs actions futures et par suite mieux cibler l'envoi de la publicité aux personnes pouvant être intéressées par le produit. Les échanges entre ces deux partenaires se dérouleraient de la manière suivante :

1) Sazri procède à une analyse parallèle de ses stocks et de ses tendances d'achats. Le résultat de cette analyse fournit la fiche du produit dont la vente peut être optimisée ;
2) L'entreprise de Marketing reçoit la fiche de ce produit et la catégorise selon ses propres critères (i.e. des bottes pour les femmes de plus de 30 ans, style classique ou non, etc.) ;
3) À l'issue de l'étape de la catégorisation, l'entreprise Sazri envoie son fichier clients ;

4) Lors de la procédure d'inscription sur le site de Sazri, cette dernière propose à ses clients la possibilité de s'inscrire soit en remplissant un formulaire, soit via leur compte réseau social Facebook ou Google +. L'entreprise de marketing profite de cet avantage afin d'élargir le fichier clients par le parcours du graphe social :
   - Si le fichier contient des inscriptions via un réseau social, l'entreprise de marketing récupère leurs contacts (dans la mesure où ceci est possible) ;
   - Sinon, le fichier clients reste inchangé.
5) À partir de la liste, ou de la liste enrichie, de clients, l'entreprise de marketing effectue son analyse de comportements clients et établit la liste des clients potentiellement intéressés. La liste des courriels de ces clients est par la suite envoyée à Sazri.

L'utilisation des outils classiques de modélisation graphique de processus BPMN 2.0 nous a donné le processus schématisé par la Figure 63. Dans ce processus collaboratif, nous retrouvons les différentes activités de l'entreprise Sazri (analyse des stocks, analyse des tendances d'achats) ainsi que celles demandées pour l'entreprise de marketing (catégorisation du produit, récupération du fichier clients et son élargissement par les réseaux sociaux, analyse des contacts réseaux sociaux, et envoi des courriels des clients ciblés).

C'est la tâche du composant médiateur d'invoquer ces différentes activités. Nous retrouvons également la modélisation des étapes parallélisées (analyse des stocks et analyse des tendances d'achats) et des étapes conditionnelles (analyse des inscriptions clients et celles qui ont été faites via un réseau social passeront à l'étape élargissement du fichier clients et les autres passeront directement à l'étape d'analyse du comportement).

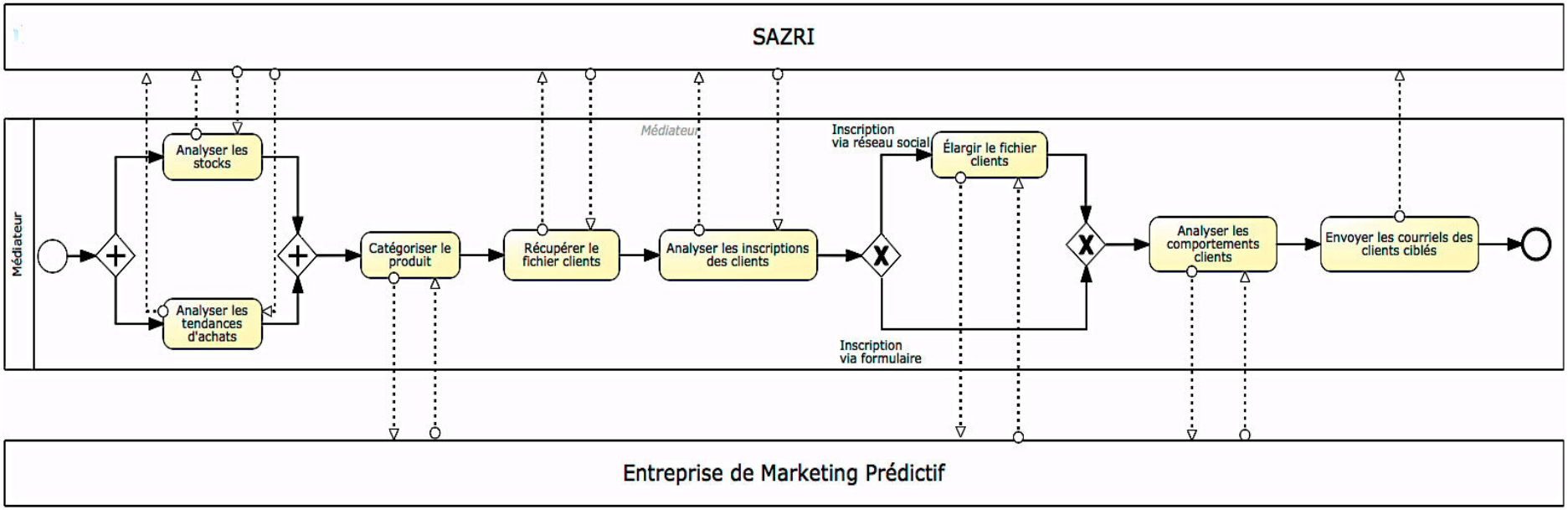

Figure 63 : *Processus métier collaboratif pour l'amélioration des ventes de Sazri par le marketing prédictif.*

Tableau 10 : *Fonctionnalités des activités du processus métier pour l'amélioration des ventes de Sazri par le marketing prédictif.*

	Activité	Fonctionnalité
Entreprise Sazri	Analyser les stocks	L'analyse des stocks consiste à étudier l'état des stocks, c'est-à-dire leur niveau, leur rotation, etc.
	Analyser les tendances d'achats	
	Récupérer le fichier clients	A l'issue de l'étape de la catégorisation du produit, le médiateur demande à l'entreprise Sazri de lui envoyer son fichier clients et le récupère.
Entreprise de marketing prédictif	Catégoriser le produit	L'entreprise de marketing devrait disposer d'un service permettant, à partir de la fiche du produit, de le catégoriser. Après traitement, ce service devrait fournir la fiche du produit catégorisé.
	Analyser les inscriptions des clients	A partir du fichier clients envoyé par l'entreprise Sazri, l'entreprise de marketing devrait disposer d'un service permettant de le parcourir et d'analyser le mode d'inscription des clients. Pour ceux qui sont inscrits via le réseau social Facebook ou Google +, le service permettant d'élargir le fichier clients devrait être invoqué. Pour ceux qui sont inscrits via le formulaire, aucun traitement à effectuer et elle passe directement à l'étape suivante, à savoir l'analyse des comportements clients.
	Élargir le fichier clients	Pour ceux qui sont inscrit via le réseau social Facebook ou Google +, l'entreprise de marketing à travers cette activité accède et analyse leur graphe social et ajoute leurs contacts et les contacts de leurs contacts et les ajoute au fichier clients initial pour obtenir le fichier cients élargi.
	Analyser les comportements clients	Cette activité prend en paramètre d'entrée le fichier clients (élargi ou initial, selon le résultat de l'activité précédente) et analyse les comportements afin d'identifier, parmi eux, ceux qui sont potentiellement intéressés par l'achat du produit sélectionné par l'entreprise Sazri. Les courriels de ces derniers seront enregistrés dans un fichier, nommé fichier clients cibles.
	Envoyer les courriels des clients ciblés	Cette activité permet d'envoyer le fichier clients cibles à l'entreprise Sazri.

Dans le Tableau 10, nous présentons les fonctionnalités de chacune des activités métier du processus collaboratif permettant l'amélioration des ventes de l'entreprise Sazri par le marketing prédictif.

Le fichier clients est ce qui est de plus précieux pour l'entreprise Sazri. Il regroupe toutes les informations concernant les clients qui lui sont essentiels et qui font vivre toutes ses activités. Il est donc indispensable, pour toute activité de l'entreprise de marketing qui utilise le fichier clients (à savoir : analyser les inscriptions des clients, élargir le fichier clients, analyser les comportements clients, et envoyer les courriels des clients ciblés), de l'accompagner d'une garantie de confidentialité et d'intégrité des données. Sazri souhaite également que ces activités proposent une procédure d'authentification.

Par ailleurs, les fêtes de Noël approchant à grands pas, l'entreprise Sazri ne souhaite pas rater cette période stratégique. Elle a donc des exigences temporelles (temps de réponse), de disponibilité, et de coût (prix). Ces exigences sont définies pour chacune des activités de l'entreprise de marketing.

Le Tableau 11 définit les exigences non-fonctionnelles de l'entreprise Sazri pour chacune des activités métier et le Tableau 12 détaille les préférences (les poids) de Sazri par rapport aux exigences non-fonctionnelles de chacune des activités métier annotées.

Tableau 11 : *Exigences non-fonctionnelles de l'entreprise Sazri.*

Activité \ Exigence non-fonctionnelle	Temps de réponse (≤)	Disponibilité (≥)	Prix (≤)	Confidentialité	Intégrité	Authentification
Catégoriser le produit	5 secondes	95%	10,00 €			
Élargir le fichier clients	40 secondes	90%	100,00 €	x	x	x
Analyser les comportements clients	1 jour	90%	100,00 €	x	x	x
Envoyer les courriels des clients ciblés	5 secondes	95%	2,00 €	x	x	x

Tableau 12 : *Préférences (poids) de l'entreprise Sazri par rapport aux exigences non-fonctionnelles.*

Activité \ Exigence non-fonctionnelle	Temps de réponse	Disponibilité	Prix	Confidentialité	Intégrité	Authentification
Catégoriser le produit	0.3	0.5	0.2			
Élargir le fichier clients	0.05	0.05	0.05	0.4	0.4	0.05
Analyser les comportements clients		0.05	0.1	0.4	0.4	0.05
Envoyer les courriels des clients ciblés	0.05	0.1		0.4	0.4	0.05

Le processus dans la Figure 63 a permis de modéliser les propriétés fonctionnelles des activités du processus métier collaboratif. Mais les questions suivantes restent ouvertes : comment modéliser les exigences non-fonctionnelles et les préférences demandées par l'entreprise Sazri ? Comment les prendre en considération pour trouver les services qui répondent à ses attentes ?

V.3.2. Déroulement de la méthodologie

Cette section a pour objectif d'illustrer l'application de notre méthodologie au cas d'étude présenté ci-dessus afin de réconcilier les activités du processus métier collaboratifs et les services techniques disponibles. Dans cette partie, nous ne re-détaillerons pas le principe de fonctionnement de chaque étape. Nous vous invitons pour cela à vous référer aux Chapitre III et Chapitre IV.

V.3.2.1 Modélisation et annotation non-fonctionnelle du processus

La première étape de notre méthodologie consiste à modéliser le processus métier décrivant la collaboration entre Sazri et l'entreprise de marketing prédictif, puis d'annoter les activités par les exigences non-fonctionnelles.

Nous reprenons le processus présenté précédemment (cf. Figure 63) et nous le modélisons avec l'outil de modélisation graphique Petals BPM-NFR. Pour ce faire, nous créons un nouveau projet métier dédié à ce cas d'étude (Sazri / MarketingPrédictif). Par la suite, à partir de la palette, nous sélectionnons l'élément BPMN 2.0 et nous faisons un glisser / déposer dans l'espace de modélisation. Dans la Figure 64, nous montrons la modélisation du médiateur dans Petals BPM-NFR.

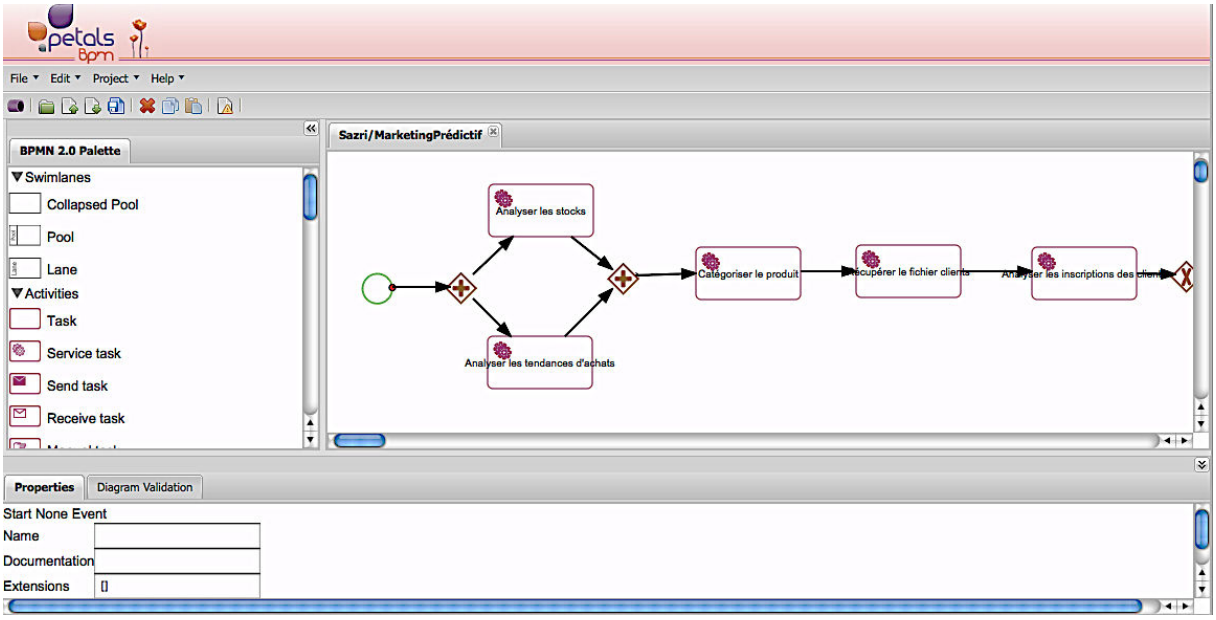

Figure 64 : *Modélisation du cas d'étude Sazri/MarketingPrédictif dans Petals BPM-NFR.*

Une fois le processus métier est modélisé, il est nécessaire d'annoter les activités de l'entreprise de marketing prédictif à l'aide d'exigences non-fonctionnelles de Sazri (qui devront être respectées lors de la phase de réconciliation non-fonctionnelle). Pour cela, nous utilisons l'interface dédiée à l'annotation non-fonctionnelle dans l'outil graphique Petals BPM-NFR et pour chacune des activités de l'entreprise de marketing, nous renseignons les valeurs et les poids des exigences non-fonctionnelles demandés par l'entreprise Sazri.

L'annotation non-fonctionnelle illustrée dans la Figure 65 ne porte que sur l'activité « Élargir le fichier client ». L'ensemble des activités de l'entreprise de marketing prédictif a été également annoté.

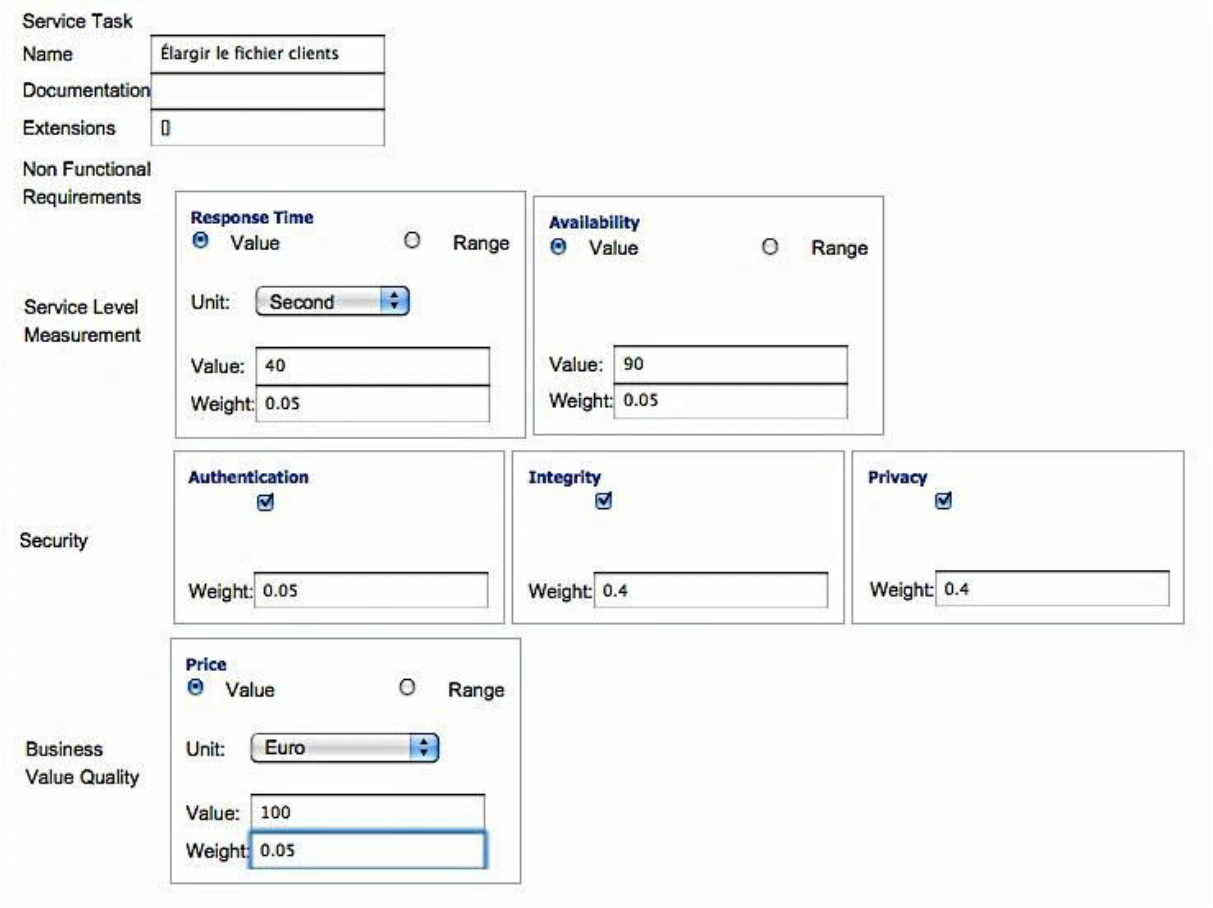

Figure 65 : *Annotation non-fonctionnelle de l'activité « Élargir le fichier clients ».*

Il est également possible d'enregistrer le processus modélisé et annoté au format BPMN2.0. Pour cela, dans la barre de menu du Petals BPM-NFR, il suffit de sélectionner *File,* puis *Export*, et enfin dans la fenêtre qui apparaît, il faut choisir le format d'export BPMN. Nous obtenons, par la suite, le fichier BPMN 2.0 correspondant au processus que nous venons de modéliser et d'annoter. Dans la Figure 66, nous présentons un extrait de ce fichier correspondant à l'activité « Élargir le fichier clients » (cf. Annexe 3 pour le fichier au format BPMN 2.0 complet). Dans cet extrait, nous retrouvons, l'ensemble des exigences non-fonctionnelles choisies ainsi que leurs valeurs. Rappelons que les exigences non-fonctionnelles de type booléen (telles que l'*Authentication*, *Integrity* et *Privacy* choisies dans notre exemple) n'apparaissent dans le BPMN 2.0 que si elles ont été sélectionnées par l'utilisateur lors de la phase d'annotation.

```
- <bpmn20:serviceTask default="_1382291745017id52" id="_1382291720000id32" name="Élargir le fichier clients">
  - <bpmn20:extensionElements>
    - <wsqdl:ServiceLevelMeasurement>
      - <wsqdl:MeasureFactor>
        - <wsqdl:ResponseTime>
          - <wsqdl:MetricValue>
              <wsqdl:Value xsi:type="xs:float">40.0</wsqdl:Value>
              <wsqdl:Type>float</wsqdl:Type>
              <wsqdl:Unit>Second</wsqdl:Unit>
            </wsqdl:MetricValue>
          </wsqdl:ResponseTime>
        - <wsqdl:Availability>
          - <wsqdl:MetricValue>
              <wsqdl:Value xsi:type="xs:float">90.0</wsqdl:Value>
              <wsqdl:Type>percent</wsqdl:Type>
            </wsqdl:MetricValue>
          </wsqdl:Availability>
        </wsqdl:MeasureFactor>
      </wsqdl:ServiceLevelMeasurement>
    - <wsqdl:BusinessValueQuality>
      - <wsqdl:ServiceCost>
        - <wsqdl:ServicePrice>
            <wsqdl:Price unit="Euro">100</wsqdl:Price>
          </wsqdl:ServicePrice>
        </wsqdl:ServiceCost>
      </wsqdl:BusinessValueQuality>
    - <wsqdl:Security>
        <wsqdl:Property name="Authentication"> </wsqdl:Property>
      </wsqdl:Security>
    - <wsqdl:Security>
        <wsqdl:Property name="Integrity"> </wsqdl:Property>
      </wsqdl:Security>
    - <wsqdl:Security>
        <wsqdl:Property name="Privacy"> </wsqdl:Property>
      </wsqdl:Security>
    </bpmn20:extensionElements>
  </bpmn20:serviceTask>
```

Figure 66 : *Extrait du fichier BPMN 2.0 : activité « Élargir le fichier clients ».*

V.3.2.2 Réconciliation métier / technique

À l'issue de l'étape de modélisation et d'annotation non-fonctionnelle du processus métier collaboratif Sazri / Marketing Prédictif, l'étape de réconciliation entre les activités métier de l'entreprise et les services techniques disponibles dans notre registre de gouvernance SOA (EasierGov-NFR) se présente.

La démarche décrite dans cette section correspond à la réconciliation de l'activité « Élargir le fichier clients ». Elle est identique pour toutes les autres activités métier du processus collaboratif. Il s'agit dans un premier temps de faire appel à l'outil EasierSBS pour réaliser la réconciliation fonctionnelle. Par la suite, pour chacun des services / compositions de services fournis par cette réconciliation, nous analysons leurs templates SLA afin de récupérer les valeurs des propriétés non-fonctionnelles. À partir de ces valeurs, nous construisons notre matrice : services candidats / propriétés

non-fonctionnelles demandées. Enfin, nous effectuons la réconciliation non-fonctionnelle pour sélectionner et classer les services qui se rapprochent des exigences de l'activité métier.

- *Réconciliation fonctionnelle*

À partir de notre activité métier modélisée et annotée dans Petals BPM-NFR, nous appliquons un premier filtre fonctionnel (grâce à l'outil EasierSBS) afin de trouver les services (parmi ceux qui sont publiés dans EasierGov) qui répondent fonctionnellement aux besoins de cette activité. Dans la Figure 67, les résultats de l'application du filtre pour l'activité « Élargir le fichier clients » sont présentés.

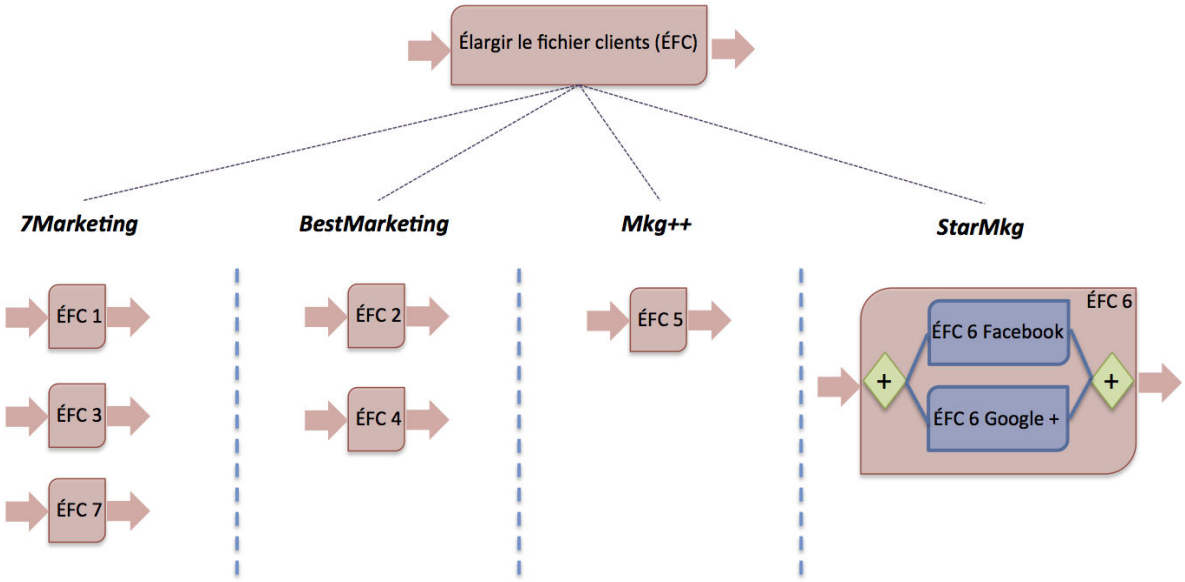

Figure 67 : *Réconciliation fonctionnelle : activité « Élargir le fichier clients ».*

Ainsi, comme nous le montre la Figure 67, il existe quatre entreprises candidates : 7Marketing, BestMarketing, Mkg++ et StarMkg. Les trois premières ont respectivement trois, deux et un services unitaires qui répondent à la fonctionnalité d'élargissement du fichier clients par le parcours des réseaux sociaux Facebook et Google+. Quant à la quatrième (StarMkg), elle n'en possède aucun mais de dispose de deux services distincts permettant de traiter cette fonctionnalité séparément (ÉFC 6 Facebook et ÉFC 6 Google +). Le premier offre la possibilité d'élargir le fichier clients par le parcours de Facebook et le deuxième s'occupe de Google +. La composition de ces deux services et leur exécution en parallèle aboutit au même résultat demandé.

- *Gestion des propriétés non-fonctionnelles*

Cette étape consiste à créer la matrice « services et compositions de services candidats ». Afin de réaliser celle-ci, pour chacun des services unitaires et des services de la composition nous analysons leurs templates SLA publiés dans EasierGov-NFR. Nous présentons dans la Figure 68 des extraits du

template SLA du service ÉFC1 contenant les propriétés non-fonctionnelles demandées au niveau métier (cf. Annexe 3 pour voir l'intégralité du template SLA).

```
<wsag:GuaranteeTerm wsag:Name="ÉlargirLeFichierClientsGuarantee">
    <wsag:ServiceLevelObjective>
        <wsag:KPITarget>
            <wsag:KPIName>Response Time</wsag:KPIName>
            <wsag:CustomServiceLevel
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name="Response Time"
                operator="&amp;lt;">
                <wsqdl:MetricValue>
                    <wsqdl:Value>50</wsqdl:Value>
                    <wsqdl:Unit>second</wsqdl:Unit>
                </wsqdl:MetricValue>
            </wsag:CustomServiceLevel>
        </wsag:KPITarget>
    </wsag:ServiceLevelObjective>
</wsag:GuaranteeTerm>
```

```
<wsag:GuaranteeTerm wsag:Name="ÉlargirLeFichierClientsGuarantee">
    <wsag:ServiceLevelObjective>
        <wsag:KPITarget>
            <wsag:KPIName>Price</wsag:KPIName>
            <wsag:CustomServiceLevel
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name="Price"
                operator="&amp;lt;">
                <wsqdl:ServiceCost>
                    <wsqdl:ServicePrice>
                        <wsqdl:Price>90</wsqdl:Price>
                        <wsqdl:Unit>euros</wsqdl:Unit>
                    </wsqdl:ServicePrice>
                </wsqdl:ServiceCost>
            </wsag:CustomServiceLevel>
        </wsag:KPITarget>
    </wsag:ServiceLevelObjective>
</wsag:GuaranteeTerm>
```

```
<wsag:GuaranteeTerm wsag:Name="ÉlargirLeFichierClientsGuarantee">
    <wsag:ServiceLevelObjective>
        <wsag:KPITarget>
            <wsag:KPIName>Privacy</wsag:KPIName>
            <wsag:CustomServiceLevel
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name="Privacy">
                <wsqdl:Security>
                    <wsqdl:Prperty name="Privacy"></wsqdl:Prperty>
                </wsqdl:Security>
            </wsag:CustomServiceLevel>
        </wsag:KPITarget>
    </wsag:ServiceLevelObjective>
</wsag:GuaranteeTerm>
```

```
<wsag:GuaranteeTerm wsag:Name="ÉlargirLeFichierClientsGuarantee">
    <wsag:ServiceLevelObjective>
        <wsag:KPITarget>
            <wsag:KPIName>Integrity</wsag:KPIName>
            <wsag:CustomServiceLevel
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name="Integrity">
                <wsqdl:Security>
                    <wsqdl:Prperty name="Integrity">
                    </wsqdl:Prperty>
                </wsqdl:Security>
            </wsag:CustomServiceLevel>
        </wsag:KPITarget>
    </wsag:ServiceLevelObjective>
</wsag:GuaranteeTerm>
```

Figure 68 : *Extraits du template SLA du service ÉFC 1 de l'entreprise 7Marketing.*

Comme nous pouvons le remarquer dans la matrice ci-dessous, la propriété de l'authentification n'est pas assurée par le service ÉFC 1 car elle n'apparaît pas dans son template SLA (cf. Annexe 3).

Pour chacun des templates SLA, nous récupérons les valeurs des propriétés non-fonctionnelles (en nous limitant seulement à celles qui sont demandées par l'activité « Élargir le fichier clients », à savoir : *Response Time*, *Availability*, *Price*, *Privacy*, *Integrity*, et *Authentication*) et nous construisons notre matrice. Ainsi, nous obtenons la matrice suivante :

Services / Compositions de services Candidats =		Response Time	Availability	Price	Privacy	Integrity	Authentication
	ÉFC 1	50 secondes	90 %	90 €	1	1	0
	ÉFC 2	35 secondes	95 %	70 €	1	1	1
	ÉFC 3	40 secondes	30 %	50 €	0	0	1
	ÉFC 4	45 secondes	91 %	100 €	1	1	0
	ÉFC 5	40 secondes	90 %	100 €	1	1	1
	ÉFC 6	ÉFC 6 Facebook: 45 secondes	100 %	45 €	1	1	1
		ÉFC 6 Google +: 30 secondes	95 %	55 €	1	1	1
	ÉFC 7	36 secondes	100 %	85 €	0	0	0

➢ *Réconciliation non-fonctionnelle*

À cette étape de notre démarche, nous avons d'une part les exigences non-fonctionnelles de l'activité métier, et d'autre part la matrice « *candidats / propriétés non-fonctionnelles* ». Nous la terminons par une phase de réconciliation non-fonctionnelle. Comme nous avons pu le voir dans la Figure 65, nous avons deux types de réconciliations : (i) une réconciliation 1-1 (pour les trois premières entreprises) et (ii) une réconciliation 1-N (pour le cas de l'entreprise StarMkg).

Nous commençons notre démarche par le traitement de la réconciliation 1-N (composition de service ÉFC 6) afin de la rendre de type 1-1. Pour y parvenir, nous commençons par faire une première étape de classification des propriétés non-fonctionnelles demandées au niveau métier.

❖ *Étape 1 : Classification des valeurs des exigences non-fonctionnelles*

Lors de la modélisation et l'annotation au niveau métier, six exigences non-fonctionnelles ont été choisies par l'architecte métier de l'entreprise Sazri :

- Response Time : une propriété non-fonctionnelle à minimiser de type valeur temporelle ;
- Availability : une propriété non-fonctionnelle à maximiser de type valeur numérique sous forme de pourcentage ;
- Price : une propriété non-fonctionnelle à minimiser de type monétaire ;

- Privacy : une propriété non-fonctionnelle à maximiser de type booléen ;
- Integrity : une propriété non-fonctionnelle à maximiser de type booléen ;
- Authentication : une propriété non-fonctionnelle à maximiser de type booléen.

❖ *Étape 2 : Transformation de la réconciliation 1-N en une réconciliation 1-1*

Nous appliquons les règles de transformation (cf. Chapitre 4, sous-section IV.5.2) et nous obtenons le vecteur suivant :

	Response Time	Availability	Price	Privacy	Integrity	Authentication
ÉFC 6 =	45	0,95	100	1	1	1

Dans la matrice *services-compositions de services candidates* obtenue, nous remplaçons les deux vecteurs lignes qui correspondaient aux deux services ÉFC Facebook et ÉFC Google + par le vecteur ÉFC 6 que nous venons d'obtenir par la transformation. Ainsi, nous obtenons la matrice des services candidats suivante :

		Response Time	Availability	Price	Privacy	Integrity	Authentication
	ÉFC 1	50 secondes	90 %	90 €	1	1	0
	ÉFC 2	35 secondes	95 %	70 €	1	1	1
	ÉFC 3	40 secondes	30 %	50 €	0	0	1
Services Candidats =	ÉFC 4	45 secondes	91 %	100 €	1	1	0
	ÉFC 5	40 secondes	90 %	100 €	1	1	1
	ÉFC 6	45 secondes	95 %	100 €	1	1	0
	ÉFC 7	36 secondes	100 %	85 €	0	0	0

À cette matrice nous appliquons l'algorithme de réconciliation non-fonctionnelle 1-1 en commençant par l'étape 2 (l'étape 1 de l'algorithme est la classification d'exigences non-fonctionnelles faites ci-dessus).

❖ *Étape 2 : Élimination des services candidats ayant des valeurs aberrantes*

Dans la matrice services candidats de l'activité « Élargir le fichier clients », il existe deux services qui ont au moins une valeur aberrante : (i) le service ÉFC 3 (disponibilité 30%, or l'architecte métier demande au moins 90% ; le service n'assure ni la confidentialité ni l'intégrité, or ces deux propriétés sont de poids d'exigences forts), et le service ÉFC 7 (il n'assure pas les propriétés de

confidentialité ni d'intégrité). Nous éliminons ces deux services de notre traitement de réconciliation non-fonctionnelle (leur élimination se fait de la matrice services candidats).

❖ *Étape 3 : Normalisation et centrage des services candidats par rapports aux exigences non-fonctionnelles de l'activité métier*

En suivant les différentes étapes de normalisation et de centrage décrites dans le chapitre IV, nous obtenons la matrice de services candidats normalisés et centrés par rapport aux exigences non-fonctionnelles de l'activité « Élargir le fichier clients » suivante :

Services Candidats normalisés et centrés =

	Response Time	Availability	Price	Privacy	Integrity	Authentication
ÉFC 1	-0.25	0	0.1	0	0	-1
ÉFC 2	0.125	0.0555	0.3	0	0	0
ÉFC 4	-0.125	0.0111	0	0	0	-1
ÉFC 5	0	0	0	0	0	0
ÉFC 6	-0.125	0.0555	0	0	0	-1

❖ *Étape 4 : Classement et sélection des services candidats*

À l'issue de l'étape de normalisation et centrage des services candidats, nous divisons la matrice obtenue à l'étape précédente en deux zones A et B. La zone A contient tous les services qui sont meilleurs que les exigences de l'activité métier sur toutes les propriétés non-fonctionnelles. Si nous reprenons notre cas d'étude, nous retrouvons dans cette sous-matrice les services ÉFC 2 et ÉFC 5. La zone B contient les autres services, c'est-à-dire : ÉFC 1, ÉFC 4 et ÉFC 6.

Nous appliquons l'algorithme de calcul de distance pondérée et de classement dédié à chacune de ces sous-matrices, et nous obtenons les résultats donnés par la Figure 69.

```
****************************************Step 4: Sorting of Candidate Services****************************************

Zone A : ÉFC2 ÉFC5
        Weighted Distance between ÉFC2 and Business Task Requirements: 0.005435570987654323
        Weighted Distance between ÉFC5 and Business Task Requirements: 0.0
Zone B : ÉFC1 ÉFC4 ÉFC6
        Weighted Distance between ÉFC1 and Business Task Requirements: 0.053125000000000006
        Weighted Distance between ÉFC4 and Business Task Requirements: 0.05078125
        Weighted Distance between ÉFC6 and Business Task Requirements: 0.05078125
                ÉFC6 :  Weighted Distance (+) 1.5432098765432115E-4
                ÉFC4 :  Weighted Distance (+) 6.172839506172797E-6

Candidate Services Ranking
1_ÉFC2
2_ÉFC5
3_ÉFC6
4_ÉFC4
5_ÉFC1
```

Figure 69 : *Résultats de la réconciliation non-fonctionnelle de l'activité « Élargir le fichier clients ».*

Ainsi, comme nous le montre la Figure 69, pour les deux services candidats de la zone A : le service ÉFC 2 a une distance pondérée supérieure à celle d'ÉFC 5, ce qui explique son classement en premier. En effet, la distance pondérée du service candidat ÉFC 5 est nulle, cela signifie que toutes les valeurs de ses propriétés non-fonctionnelles (uniquement celles qui ont été choisies au niveau métier) sont identiques à celles des exigences non-fonctionnelles de l'activité métier. Quant au service ÉFC 2, il répond mieux que ce qui a été demandé au niveau métier. Il a un temps de réponse inférieur, plus de disponibilité, et il est moins cher.

Pour les services de la zone B : les services ÉFC 4 et ÉFC 6 ont des distances pondérées identiques. C'est pour ça que nous avons calculé leurs distances pondérées (+) c'est-à-dire, en ne prenant en considération que leurs valeurs des propriétés non-fonctionnelles ≥ 0. Ce calcul a montré que la composition de services ÉFC 6 est plus proche des exigences non-fonctionnelles métier que le service ÉFC 4. Finalement, le service ÉFC 1 se classe en dernier car il a la plus grande distance pondérée de toute la zone B.

V.4. Conclusion

Dans ce chapitre, nous avons présenté les détails de l'implémentation réalisée. Cette implémentation porte sur (i) notre approche de modélisation et d'annotation non-fonctionnelle du processus métier collaboratif « Petals BPM-NFR » et (ii) notre démarche de gouvernance SOA

permettant la gestion des services et de leurs propriétés non-fonctionnelles, et la réconciliation non-fonctionnelle « EasierGov-NFR ».

Ce chapitre nous a également permis d'illustrer nos contributions sur un cas d'étude concernant la collaboration entre deux entreprises (où un SI de médiation gère cette collaboration). Ainsi, après avoir détaillé le cas d'étude et expliqué les diverses attentes non-fonctionnelles de l'architecte métier, nous avons montré qu'il était judicieux de disposer d'une plateforme de modélisation permettant à la fois de modéliser les activités métier graphiquement et de les annoter par des exigences non-fonctionnelles afin de répondre au mieux aux attentes métier. Ensuite, nous avons modélisé et annoté non-fonctionnellement le processus métier collaboratif en utilisant la plateforme Petals BPM-NFR et en respectant l'approche proposée dans le Chapitre III. Enfin, à partir de ce processus et en utilisant EasierGov-NFR, nous avons appliqué notre démarche (dans le respect de celle proposée dans les Chapitres III et IV) de gouvernance SOA pour effectuer la réconciliation non-fonctionnelle entre le niveau métier et le niveau technique.

Cependant, plusieurs améliorations sont possibles, notamment par rapport à l'ergonomie de nos deux prototypes. De plus, bien que nous ayons essayé d'optimiser notre démarche de gouvernance SOA, cette dernière peut être largement améliorée et étendue par l'ajout de la réconciliation N-M (plusieurs activités métier – plusieurs services).